AMENDMENT

In the Claims:

1. (Cancelled)

2. (Currently Amended) A computer-implemented method, comprising:

assigning a definition-node for one or more definition statements in an intermediate language program;

assigning a use-node for one or more use statements in the intermediate language program;

performing a memory alias analysis of the intermediate language program to partition memory accesses into equivalence classes such that any two memory accesses that reference the same storage location belong to the same equivalence class;

assigning an alias-node for one or more aliases representing the equivalence class of the memory accesses;

introducing an edge into a dependence flow graph connecting each definition-node to the alias-node corresponding to the alias representing the equivalence class to which the definition-node belongs;

introducing an edge in the dependence flow graph connecting each use-node to the alias-node corresponding to the alias representing the equivalence class to which the use-node belongs; and

performing a program analysis using the dependence flow graph by assigning, for each alias-node in the dependence flow graph, an initial value to the alias corresponding to ~~said~~the alias-node and adding the alias-node to a set of nodes;

wherein a number of the edges in the dependence flow graph is linear to a number of the nodes in the dependence flow graph, and wherein the number of edges is independent of a definition-use structure of the intermediate language program;

wherein the program analysis further comprises iteratively performing while the set of nodes is not empty:

removing a node from the set of nodes;

if the node is an alias-node, adding successors of the node in the dependence flow graph to the set of nodes; and if the node is a definition-node for a statement that defines a storage location:

determining a value for an expression to be written to the storage location;

updating the initial value based on the value of the expression; and

adding the storage location to the set of nodes.

3. (Canceled)

4. (Canceled)

5. (Canceled)

6. (Previously Presented) The computer-implemented method of claim 2, wherein the initial value comprises a set of abstract values which forms a join-complete partial order.

7. (Canceled)

8. (Currently Amended) A machine-readable medium that stores instructions, which when executed by a processor, cause the processor to perform operations comprising:

assigning a definition-node for one or more definition statements in an intermediate language program;

assigning a use-node for one or more use statements in the intermediate language program;

performing a memory alias analysis of the intermediate language program to partition memory accesses into equivalence classes such that any two memory accesses that reference the same storage location belong to the same equivalence class;

assigning an alias-node for one or more aliases representing the equivalence class of the memory accesses;

introducing an edge into a dependence flow graph connecting each definition-node to the alias-node corresponding to the alias representing the equivalence class to which the definition-node belongs;

introducing an edge in the dependence flow graph connecting each use-node to the alias-node corresponding to the alias representing the equivalence class to which the use-node belongs; and

performing a program analysis using the dependence flow graph by assigning, for each alias-node in the dependence flow graph, an initial value to the alias corresponding to ~~said~~the alias-node and adding the alias-node to a set of nodes;

wherein a number of the edges in the dependence flow graph is linear to a number of the nodes in the dependence flow graph, and wherein the number of edges is independent of a definition-use structure of the intermediate language program;

wherein the program analysis further comprises iteratively performing while the set of nodes is not empty:

removing a node from the set of nodes;

if the node is an alias-node, adding successors of the node in the dependence flow graph to the set of nodes; and

if the node is a definition-node for a statement that defines a storage location:

determining a value for an expression to be written to the storage location;

updating the initial value based on the value of the expression; and adding the storage location to the set of nodes.

9. (Canceled)

10. (Canceled)

11. (Canceled)

12. (Previously Presented) The machine-readable medium of claim 8, wherein the initial value comprises a set of abstract values which forms a join-complete partial order.

13. (Cancelled)

14. (Currently Amended) An apparatus, comprising:

a memory;

a processor coupled to the memory and having a set of instructions which when executed by the processor cause the processor to perform operations comprising:

assigning a definition-node for one or more definition statements in an intermediate language program;

assigning a use-node for one or more use statements in the intermediate language program;

performing a memory alias analysis of the intermediate language program to partition memory accesses into equivalence classes such that any two memory accesses that reference the same storage location belong to the same equivalence class;

assigning an alias-node for one or more aliases representing the equivalence class of the memory accesses;

introducing an edge into a dependence flow graph connecting each definition-node to the alias-node corresponding to the alias representing the equivalence class to which the definition-node belongs;

introducing an edge in the dependence flow graph connecting each use-node to the alias-node corresponding to the alias representing the equivalence class to which the use-node belongs; and

performing a program analysis using the dependence flow graph by assigning, for each alias-node in the dependence flow graph, an initial value to the alias corresponding to ~~said~~the alias-node and adding the alias-node to a set of nodes;

wherein a number of the edges in the dependence flow graph is linear to a number of the nodes in the dependence flow graph, and wherein the number of edges is independent of a definition-use structure of the intermediate language program;

wherein the program analysis further comprises iteratively performing while the set of nodes is not empty:

removing a node from the set of nodes;

if the node is an alias-node, adding successors of the node in the dependence flow graph to the set of nodes; and

if the node is a definition-node for a statement that defines a storage location:

determining a value for an expression to be written to the storage location;

updating the initial value based on the value of the expression; and

adding the storage location to the set of nodes.

15. (Canceled)

16. (Canceled)

17. (Canceled)

18. (Previously Presented) The apparatus of claim 14, wherein the initial value comprises a set of abstract values which forms a join-complete partial order.

19.-20. (Canceled)

21. (Currently Amended) An computer-implemented apparatus, comprising:

means for assigning a definition-node for one or more definition statements in an intermediate language program;

means for assigning a use-node for one or more use statements in the intermediate language program;

means for performing a memory alias analysis of the intermediate language program to partition memory accesses into equivalence classes such that any two memory accesses that reference the same storage location belong to the same equivalence class;

means for assigning an alias-node for one or more aliases representing the equivalence class of the memory accesses;

means for introducing an edge into a dependence flow graph connecting each definition-node to the alias-node corresponding to the alias representing the equivalence class to which the definition-node belongs;

means for introducing an edge in the dependence flow graph connecting each use-node to the alias-node corresponding to the alias representing the equivalence class to which the use-node belongs; and

means for performing a program analysis using the dependence flow graph by assigning, for each alias-node in the dependence flow graph, an initial value to the alias corresponding to ~~said~~the alias-node and adding the alias-node to a set of nodes;

wherein a number of the edges in the dependence flow graph is linear to a number of the nodes in the dependence flow graph, and wherein the number of edges is independent of a definition-use structure of the intermediate language program;

wherein the program analysis further comprises iteratively performing while the set of nodes is not empty:

removing a node from the set of nodes;

if the node is an alias-node, adding successors of the node in the dependence flow graph to the set of nodes; and

if the node is a definition-node for a statement that defines a storage location:

determining a value for an expression to be written to the storage location;

updating the initial value based on the value of the expression; and

adding the storage location to the set of nodes.

22. (Canceled)

23. (Canceled)

24. (Canceled)

25. (Canceled)

26. (Previously Presented) The apparatus of claim 21, wherein the initial value comprises a set of abstract values which forms a join-complete partial order.